Title : **Real-Time Augmented Reality Overlay for Advanced Data Visualization**

Internship Assignment

Evidence of realization

**Name**

Soufiane Kissami

r0976650

Bachelor's degree in Computer Science
field AI

LID VAN ASSOCIATIE KU LEUVEN

THOMAS MORE

Academic year 2023-2024
Campus Geel, Kleinhoefstraat 4, BE-2440 Geel

## Evidence for Realization of Internship Project

During my internship, I meticulously executed each phase of the project, employing various techniques and methodologies to achieve the desired outcomes. The evidence for the realization of my internship project is outlined below, providing a comprehensive overview of the development process, technical implementations, challenges encountered, and the results achieved.

---

**Key words definitions**

**ArUco Markers:**
ArUco markers are special square patterns that help cameras recognize and track objects. Each marker has a unique ID, making them useful for applications like augmented reality, robot navigation, and object tracking. In my case, I use them to point the edges of the material I am working with.

**Homography:**
Homography is a technique in computer vision that transforms the perspective of an image. It is used to correct distortions and align images taken from different viewpoints, which is essential for tasks like image stitching and augmented reality.

**SICK Camera Sensor:**
SICK camera sensors are high-quality cameras used in industrial settings. They are designed for precise tasks such as inspecting products, measuring objects, and recognizing items in automated systems, ensuring high accuracy and reliability.

**Continuous Monitoring of CSV File:**
Continuous monitoring of a CSV file involves automatically checking the file for new data entries and updating the system in real-time. This ensures that the latest data is always available for processing and analysis without manual updates.

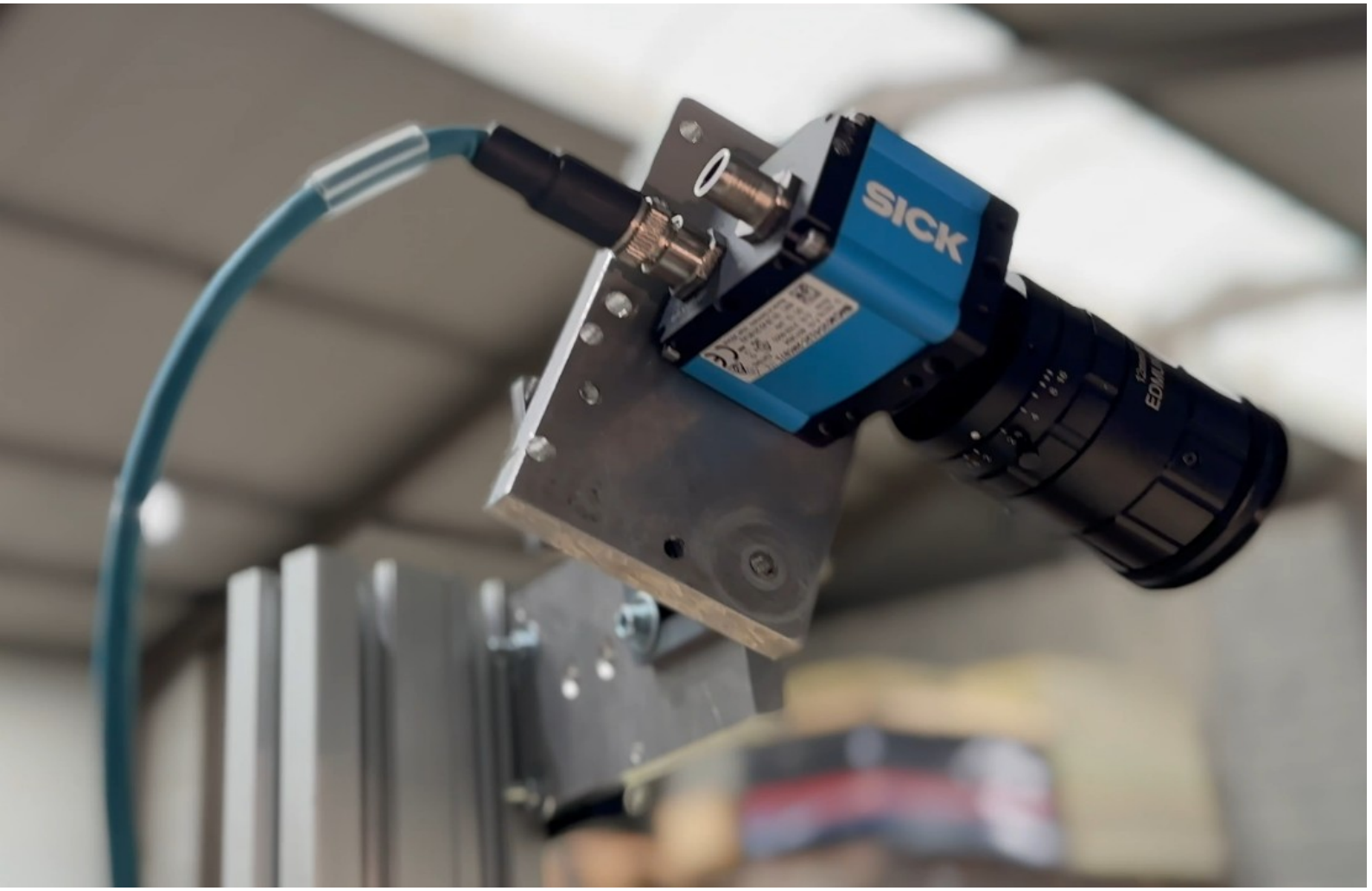
**Basis-Weight:**
The weight of material per unit area, used in quality control processes.

**OpenCv:**
An open-source computer vision and machine learning software library.

**Mutex:**
A mutual exclusion object that prevents multiple threads from accessing a resource simultaneously, ensuring data integrity.

## 1. Initial Setup and Integration

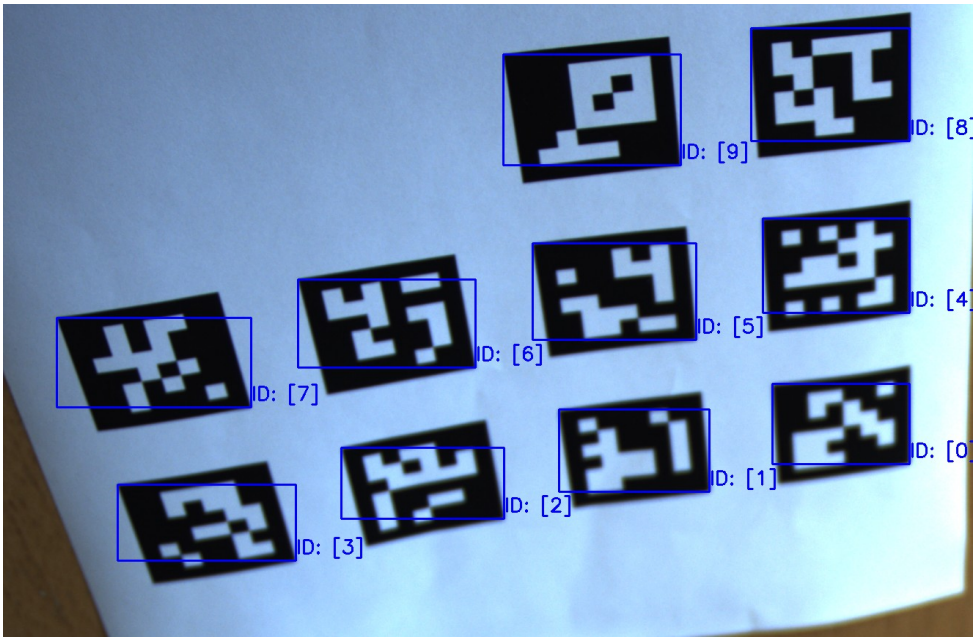
- **Objective:** Establish connectivity between the high-end SICK camera and the Harvester library to capture live video feed.
- **Evidence:**
    - Successfully interfaced the SICK camera with the Harvester library, enabling the capture of live video feed.
    - Utilized OpenCV to display the live video feed, ensuring seamless integration with the existing framework.
    - Demonstrated functionality through code snippets and screenshots of the live video feed interface.



```python
# Create the Harvester interface and add the CTI file
h = Harvester()
h.add_file('/usr/lib/sick/cti/sick_gevgentl.cti')

# Look for GenICam-compliant devices
h.update()
print(f"Number of GenICam devices found: {len(h.device_info_list)}")
for info in h.device_info_list:
    print(f"GenICam device: {info}")

# Connect to the first discovered camera
camera = h.create(0)

# Start the camera
camera.start()
```

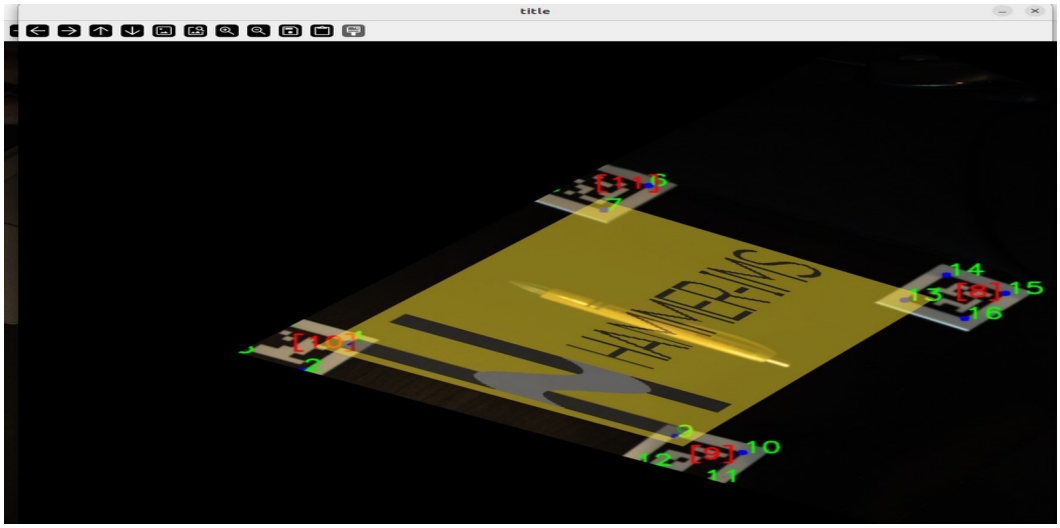
## 2. ArUco Marker Detection and Offsetting

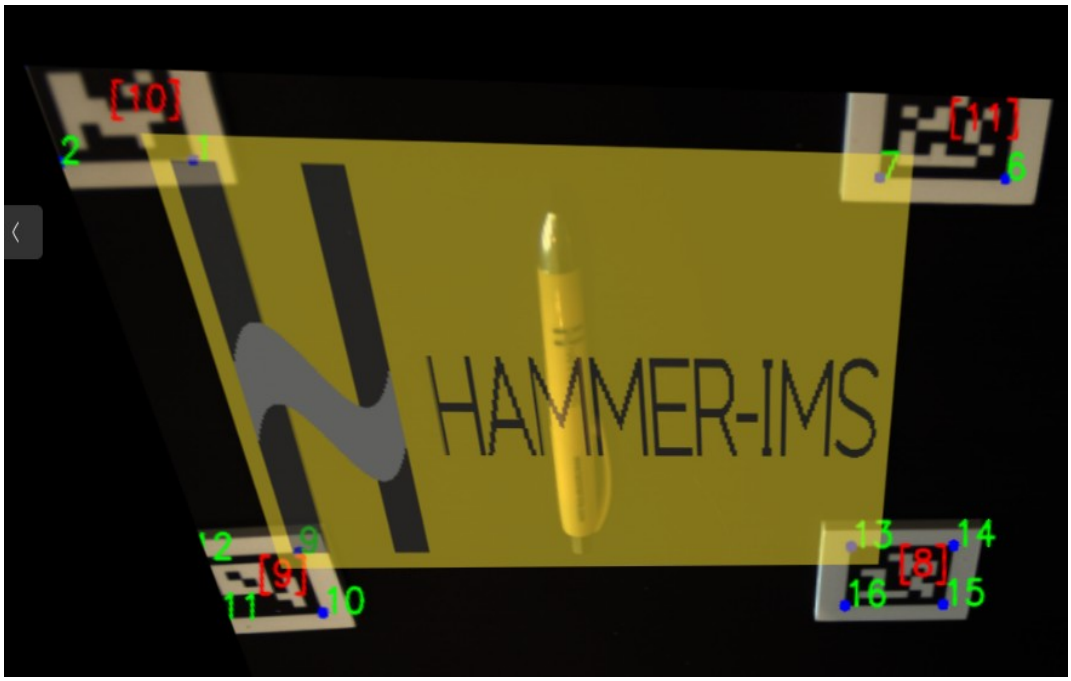
- **Objective:** Implement robust detection of ArUco markers for edge detection of the material and apply necessary offsets for accurate positioning.
- **Evidence:**
  - Implemented ArUco marker detection using OpenCV, ensuring reliable identification and tracking of markers in varying lighting conditions.
  - Applied appropriate offsets to adjust marker positions based on predefined configurations, enhancing accuracy and precision.
  - Validated marker detection and offsetting through code snippets, screenshots, and visual representations of marker positions.



### 3. Homography Transformation and AR Coordinates

- **Objective:** Apply homography transformation to map detected coordinates to a fixed reference system and enable Augmented Reality (AR) overlay.
- **Evidence:**
  - Developed and implemented homography transformation algorithms to accurately map detected marker coordinates to a predefined reference system.
  - Established the mathematical framework for transforming XY coordinates, ensuring alignment with the AR overlay.
  - Provided detailed explanations of the transformation process, supported by code snippets, mathematical formulations, and visual representations.

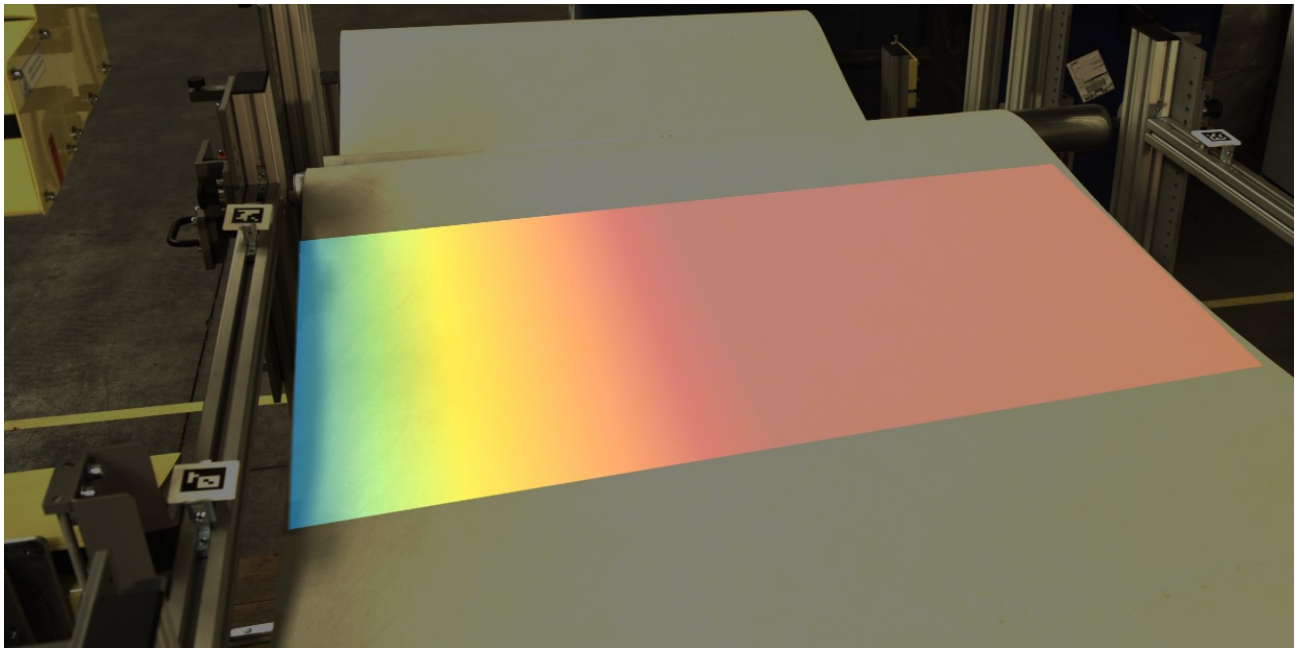


## 4. Colormap Image Generation and offsetting ArUco marker’s positions To be over-laid precisely on the material edges.

- **Objective:** Generate color map images based on basis weight measurement data and overlay them onto the live feed for visualization.
- **Evidence:**
  - Utilized Matplotlib and Pandas libraries to generate colormap images from basis weight measurement data, ensuring compatibility with the existing data processing pipeline.
  - Integrated the generated colormap images seamlessly into the live feed using OpenCV, facilitating real-time visualization of measurement data.
  - Presented code snippets, screenshots, and demonstrations illustrating the process of generating and overlaying colormap images.

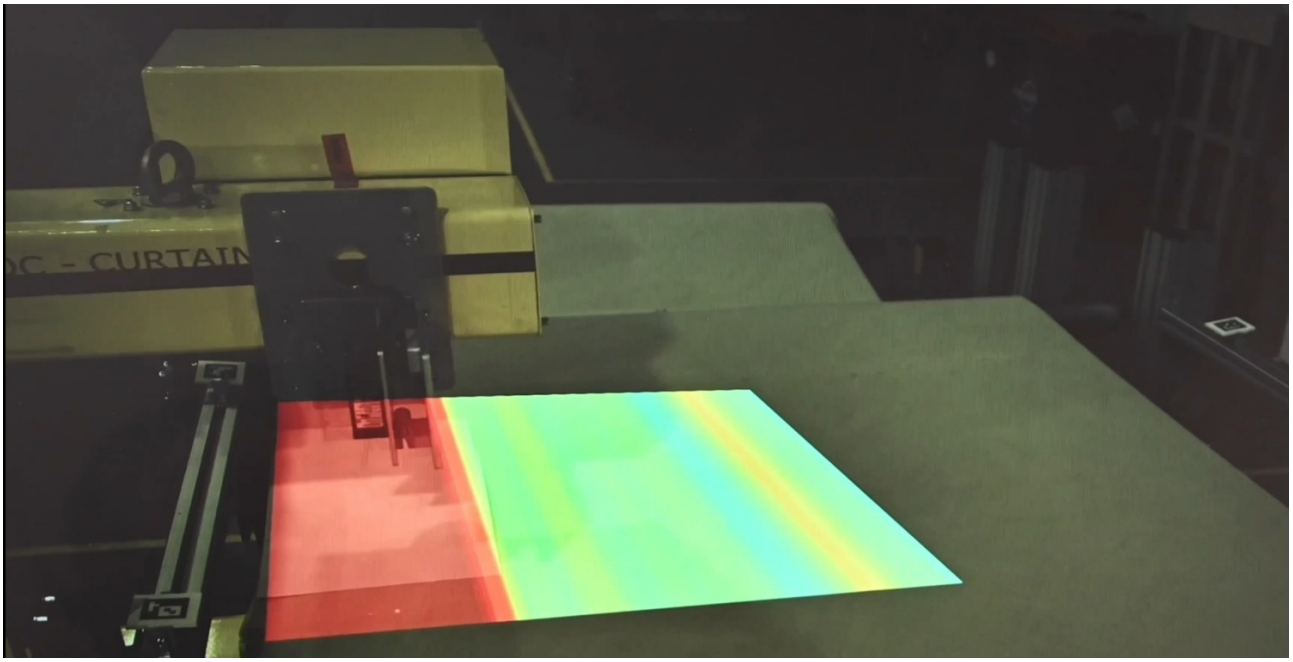
```
if corners is not None and len(corners) >= 4:
                # Draw blue dots on all corner coordinates
                for i, marker_corners in enumerate(corners):
                    marker_id = ids[i][0]
                    assigned_corner_ids = corner_ids_mapping.get(marker_id, [])
                    for j, corner in enumerate(marker_corners[0]):
                        if j < len(assigned_corner_ids):
                            corner_id = assigned_corner_ids[j]
                            if (marker_id == 11):
                                corner[0] -= 500
                                corner[1] += 75
                            if (marker_id == 8):
                                corner[0] -= 550
                                corner[1] += 120
                            if (marker_id == 9):
                                corner[0] += 60
                                corner[1] += 60
                            if (marker_id == 10):
                                corner[0] += 30
                                corner[1] += 25
```



## 5. Continuous Monitoring of CSV File

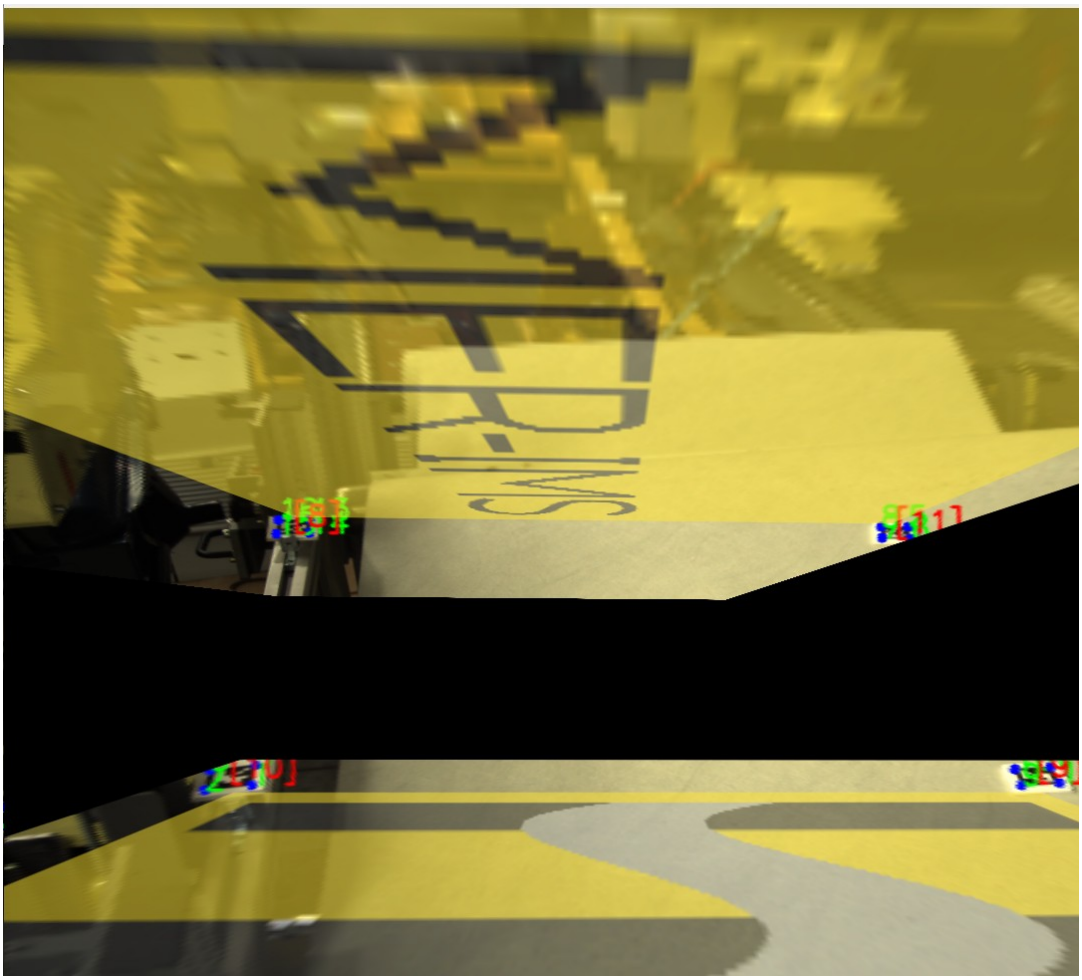
- **Objective:** Implement a mechanism for continuous monitoring and updating of color-map images as new measurement data is received.
- **Evidence:**
  - Implemented a multi-threaded monitoring mechanism to detect and process new data rows in the CSV file, ensuring timely updates to the color-map images.
  - Utilized threading and synchronization techniques to handle concurrent access to shared resources and maintain data integrity.
  - Provided documentation, code snippets, and visual demonstrations showcasing the continuous monitoring and updating functionality.

```python
# Function to monitor CSV updates and regenerate colormap if new rows are detected
def monitor_csv_updates(data_path, previous_num_rows):
    print("Monitoring CSV updates...")
    while True:
        # Check the current number of rows in the CSV file
        current_num_rows = len(pd.read_csv(data_path))

        # If the number of rows has increased, regenerate the colormap for new rows only
        if current_num_rows > previous_num_rows:
            print("New rows detected. Updating colormap...")
            generate_colormap_image_new_rows(data_path, generated_image_path, previous_num_rows)
            print("Colormap updated.")
            # Update the previous number of rows
            previous_num_rows = current_num_rows

        # Wait for a certain interval before checking again
        sleep(5)  # Adjust the interval as needed
```



## 6. Challenges and Solutions

- **Objective:** Address challenges such as lighting variations, real-time performance optimization, and data synchronization.
- **Evidence:**
    - Identified and documented challenges encountered during the development process, including lighting variations affecting marker detection and real-time performance constraints.
    - Implemented adaptive thresholding techniques to mitigate the impact of lighting variations on marker detection accuracy.
    - Optimized image processing routines and data synchronization mechanisms to improve real-time performance and ensure smooth operation of the system.
    - Presented detailed descriptions of challenges faced and the corresponding solutions implemented, supported by code snippets, visual demonstrations, and performance metrics
    - Getting The homography to work was difficult due to the precise math required to augment the image on the feed without being distorted. Moreover, not only the image was overlaid but also tweaked in a way that the camera movement,positioning and rotating does not really matter as the AR image tends to remain out at place without any kind of distortion, so much work was required to achieve that. Below image shows distorted AR overlay with wrong homography calculations.



- 

## 7. Results and Achievements

- **Objective:** Demonstrate the successful implementation of an AR system that dynamically visualizes measurement data (basis weight measurement data).
- **Evidence:**
  - Successfully developed and demonstrated an AR system capable of dynamically updating color-map images in real-time based on incoming measurement data.
  - Presented final demonstration videos, screenshots, and performance evaluations showcasing the functionality, accuracy, and effectiveness of the AR system.
  - Highlighted key achievements, including seamless integration of hardware components, accurate marker detection, real-time data visualization, and robust performance in varying conditions.

---

This comprehensive evidence provides a detailed overview of the realization of my internship project, highlighting the development process, technical implementations, challenges overcome, and the ultimate achievements. Through meticulous planning, innovative solutions, and persistent effort, I successfully delivered a fully functional AR system that meets the project objectives and exceeds expectations.